CLAIM AMENDMENTS

1. (Currently Amended) A traffic management processor for processing a plurality of different traffic flows on a per-flow basis, each traffic flow including any number of packets transmitted from the same source address to the same destination address and each packet including a flow identification (ID) indicating to which traffic flow the packet belongs, comprising:

means for tracking each packet according to its flow ID, wherein the means for tracking comprises a content addressable memory (CAM) device having a plurality of rows, each for storing the flow ID and a most recently received bit for a corresponding packet, wherein the most recently received bit indicates whether the corresponding packet is the most recently received packet for its traffic flow; and

means for scheduling each packet according to its flow ID.

2. (Currently Amended) The traffic management processor of Claim 1, wherein the means for tracking further comprises:

~~a content address memory (CAM) device having a plurality of rows, each for storing the flow ID for a corresponding packet~~match flag logic having inputs coupled to the rows of the CAM device and having an output coupled to the means for scheduling.

3. (Currently Amended) The traffic management processor of Claim ~~2~~1, wherein the flow ID is independent of a per-hop behavior selection~~the CAM device further comprises a plurality of most recently received bits, each indicating whether a corresponding packet is the most recently received packet for its traffic flow~~.

4. (Currently Amended) The traffic management processor of Claim ~~2~~1, wherein the ~~content address memory (CAM)~~CAM device is configured to compare the flow ID of an incoming packet with the flow IDs stored in the CAM device to generate a match flag indicating whether the incoming packet is part of a new traffic flow.

5. (Original) The traffic management processor of Claim 4, wherein the means for scheduling calculates a departure time for the incoming packet relative to the packet's arrival time if the match flag is not asserted and calculates the departure time for the incoming packet relative to the departure time of the previously received packet of the same flow if the match flag is asserted.

6. (Original) The traffic management processor of Claim 5, wherein the means for scheduling comprises:

a departure time calculator (DTC) circuit for generating the departure times; and

a departure time prioritizer coupled to the DTC circuit and for determining which of the departure times is the earliest.

7. (Currently Amended) The traffic management processor of Claim 6, wherein the departure time prioritizer comprises:

a table having a plurality of rows, each row for storing the departure time for a corresponding packet and each row coupled to a corresponding row of the CAM device; and

compare logic coupled to the table, the compare logic configured to compare the departure times with each other to determine which row contains the earliest departure time.

8. (Original) The traffic management processor of Claim 7, further comprising:

a priority encoder coupled to the compare logic, the priority encoder generating an address of the row in the table that contains the earliest departure time.

9. (Original) The traffic management processor of Claim 1, further comprising:

means for independently policing each of the traffic flows.

10. (Previously Presented) The traffic management processor of Claim 9, wherein the means for independently policing comprises:

a parameter table having a plurality of rows, each for storing one or more flow parameters for a queued packet; and

policing logic coupled to the parameter table, the policing logic for generating a packet accept flag for an incoming packet in response to one or more flow parameters corresponding to a queued packet that has the same flow ID as the incoming packet.

11. (Previously Presented) The traffic management processor of Claim 10, wherein the policing logic comprises:

means for accessing a packet size parameter for the incoming packet;

means for accessing the one or more flow parameters from the parameter table for a previously queued packet that has the same flow ID as the incoming packet;

means for calculating a bucket size parameter using the one or more flow parameters; and

means for comparing the bucket size parameter with the packet size parameter to generate the packet accept flag.

12. (Currently Amended) A traffic management processor for managing a number of traffic flows each including one or more packets, wherein the packets of each flow are transmitted from the same source address to the same destination address, comprising:

a content addressable memory (CAM) device having a plurality of rows, each row for storing a flow identification (ID) and a most recently received bit for a corresponding packet, wherein the flow ID ~~indicating~~ indicates to which traffic flow the packet belongs and wherein the most recently received bit indicates whether the corresponding packet is the most recently received packet for its traffic flow;

a departure time table including a plurality of rows, each coupled to a corresponding row of the CAM device and configured to store a departure time for the corresponding packet; and compare logic having inputs coupled to corresponding rows of the departure time table, the compare logic for comparing the departure times with each other to determine which departure time is the earliest.

13. (Original) The traffic management processor of Claim 12, further comprising a priority encoder coupled to the compare logic, the priority encoder generating an address of the row in the departure time table that contains the earliest departure time.

14. (Currently Amended) The traffic management processor of Claim ~~13~~12, wherein the flow ID is independent of a per-hop behavior selection~~each row of the CAM device includes a most recently received bit that indicates whether the corresponding packet is the most recently received packet for its traffic flow~~.

15. (Currently Amended) The traffic management processor of Claim ~~14~~13, wherein the priority encoder is configured to generate a next free address in the CAM device in response to the most-recently received bits.

16. (Original) The traffic management processor of Claim 12, wherein the CAM device is configured to compare a flow ID received for an incoming packet with the flow ID's stored in the CAM device.

17. (Original) The traffic management processor of Claim 16, further comprising:

match logic having a plurality of inputs, each coupled to a corresponding row of the CAM device, the match logic generating a match flag in response to match conditions in the CAM device; and

a departure time calculator (DTC) circuit having an input to receive the match flag.

18. (Original) The traffic management processor of Claim 17, wherein the

DTC circuit calculates a departure time for the incoming packet relative to the departure time of a previously received packet of the same traffic flow if the match flag is asserted.

19. (Previously Presented) The traffic management processor of Claim 17, wherein the DTC circuit calculates a departure time for the incoming packet relative to the packet's arrival time if the match flag is not asserted.

20. (Original) The traffic management processor of Claim 12, further comprising:

a parameter table having a plurality of rows, each coupled to corresponding rows of the CAM device and the departure time table, each row of the parameter table for storing one or more flow parameters for a corresponding queued packet; and

policing logic coupled to the parameter table, the policing logic determining whether to accept or reject an incoming packet in response to one or more flow parameters selectively provided by the parameter table.

21. (Previously Presented) The traffic management processor of Claim 20, wherein the policing logic comprises:

means for accessing a packet size parameter for the incoming packet;

means for accessing one or more flow parameters from the parameter table for a previously received packet of the same traffic flow;

means for calculating a bucket size parameter using the one or more flow parameters; and

means for comparing the bucket size parameter with the packet size parameter to generate a packet accept flag for the incoming packet.

22. (Currently Amended) A method for processing a number of different traffic flows on a per-flow basis, each traffic flow including one or more packets transmitted from the same source address to the same destination address, comprising:

receiving an incoming packet, wherein the incoming packet includes a flow identification (ID) indicating to which traffic flow the incoming packet belongs;

~~determining which traffic flow the incoming packet belongs to~~

comparing the flow ID of the incoming packet with the flow ID's of previously queued packets using a content addressable memory (CAM) device configured to store the flow ID's for corresponding packets;

selectively asserting a match flag in response to the comparing; and

scheduling the incoming packet for departure according to which traffic flow the packet belongs.

23. (Currently Amended) The method of Claim 22, wherein the flow ID is independent of a per-hop behavior selection~~the incoming packet includes a flow identification (ID) indicating to which traffic flow the incoming packet belongs~~.

24. (Canceled)

25. (Currently Amended) The method of Claim ~~24~~22, wherein the scheduling comprises:

calculating a departure time for the incoming packet relative to the departure time of a previously received packet of the same traffic flow if the match flag is asserted; and

calculating a departure time for the incoming packet relative to the packet's arrival time if the match flag is not asserted.

26. (Original) The method of Claim 25, wherein the scheduling further comprises:

comparing the departure times of the packets with each other to determine which departure time is the earliest; and

transmitting the packet that has the earliest departure time.

27. (Original) The method of Claim 22, further comprising:

storing a most recently received bit for each packet.

28. (Original) The method of Claim 27, further comprising:

asserting the most-recently received bit for the incoming packet; and

de-asserting the most-recently received bit of a previously received packet of the same traffic flow as the incoming packet.

29. (Original) The method of Claim 27, further comprising:

storing the departure times for all packets together in a departure time table.

30. (Original) The method of Claim 29, further comprising:

selectively deleting entries from the table in response to the most recently received bit.

31. (Original) The method of Claim 22, further comprising:

policing the incoming packet for acceptance according to which traffic flow the packet belongs.

32. (Original) The method of Claim 31, wherein each traffic flow is independently policed using a leaky bucket technique.

33. (Original) The method of Claim 31, wherein the policing comprises, for the incoming packet:

accessing a packet size parameter for the incoming packet;

accessing a bucket size parameter for the incoming packet's traffic flow, the bucket size indicating the amount of data that may be accepted from the incoming packet's traffic flow during a sample period;

comparing the bucket size parameter to the packet size parameter; and

accepting the incoming packet if the packet size parameter is less than the bucket size parameter.

34. (Original) The method of Claim 33, further comprising:

decreasing the bucket size parameter by an amount of the packet size parameter if the incoming packet is accepted.

35. (Previously Presented) The traffic management processor of Claim 1, wherein the different traffic flows are not aggregated.

36. (Previously Presented) The traffic management processor of Claim 12, wherein the traffic flows are not aggregated.

37. (Previously Presented) The traffic management processor of Claim 12, wherein the traffic flows are managed on a per-flow basis.

38. (Previously Presented) The method of Claim 22, wherein the traffic flows are not aggregated.